AdaFocusV3: On Unified Spatial-temporal Dynamic Video Recognition

Yulin Wang[1⋆], Yang Yue[1⋆], Xinhong Xu[1], Ali Hassani[2], Victor Kulikov[3], Nikita Orlov[3], Shiji Song[1], Humphrey Shi[2,3†], and Gao Huang[1,4†]

[1] Department of Automation, BNRist, Tsinghua University
[2] University of Oregon
[3] Picsart AI Research (PAIR)
[4] Beijing Academy of Artificial Intelligence (BAAI)
{wang-yl19, ley18}@mails.tsinghua.edu.cn, shihonghui3@gmail.com, gaohuang@tsinghua.edu.cn

Abstract. Recent research has revealed that reducing the *temporal* and *spatial* redundancy are both effective approaches towards efficient video recognition, *e.g.*, allocating the majority of computation to a task-relevant subset of frames or the most valuable image regions of each frame. However, in most existing works, either type of redundancy is typically modeled with another absent. This paper explores the unified formulation of spatial-temporal dynamic computation on top of the recently proposed AdaFocusV2 algorithm, contributing to an improved AdaFocusV3 framework. Our method reduces the computational cost by activating the expensive high-capacity network only on some small but informative 3D video cubes. These cubes are cropped from the space formed by frame height, width, and video duration, while their locations are adaptively determined with a light-weighted policy network on a per-sample basis. At test time, the number of the cubes corresponding to each video is dynamically configured, *i.e.*, video cubes are processed sequentially until a sufficiently reliable prediction is produced. Notably, AdaFocusV3 can be effectively trained by approximating the non-differentiable cropping operation with the interpolation of deep features. Extensive empirical results on six benchmark datasets (*i.e.*, ActivityNet, FCVID, Mini-Kinetics, Something-Something V1&V2 and Diving48) demonstrate that our model is considerably more efficient than competitive baselines.

Keywords: efficient video analysis, dynamic neural networks, action recognition.

1 Introduction

Modern deep networks have reached or even surpassed human-level performance on large-scale video recognition benchmarks [46,15,4,23,13,12,1]. Such a remarkable success is in general fueled by the rapid development of large models with

⋆ Equal contributions. †Corresponding Authors.

high computational demands at inference time. However, the practical application of these resource-hungry models may be challenging, despite their state-of-the-art accuracy. For example, in real-world scenarios like YouTube video recommendation [8,9,17], video-based surveillance [7,5] and content-based searching engines [27], deploying computationally intensive networks significantly increases power consumption, system latency or carbon emission, all of which should be minimized due to economic and environmental concerns.

To address this issue, a number of recent works focus on reducing the inherent *temporal* or *spatial* redundancy in video analysis. For instance, for the former, several algorithms have been proposed to dynamically identify the most informative frames and allocate the majority of computation to them, yielding improvements in the overall efficiency [61,58,18,32,41,45,36]. For the latter, the recently proposed adaptive focus networks (AdaFocus) [52,56] develop dynamic models to adaptively attend to the task-relevant regions of each video frame, which considerably reduces the computational cost without sacrificing accuracy. Although both directions have been proven feasible, how to model *spatial* and *temporal* redundancy jointly and realize highly efficient spatial-temporal dynamic computation is still an under-explored topic. The preliminary results presented by AdaFocusV2+ [56] have revealed their favorable compatibility. However, AdaFocusV2+ only considers them separately in independent modules.

In this paper, we seek to explore the unified formulation that by design aims to reduce the spatial-temporal redundancy simultaneously, and thus study whether such general frameworks can lead to a more promising approach towards efficient video recognition. To implement this idea, we propose an AdaFocusV3 network. In specific, given an input video, our method first takes a quick and cheap glance at it with a light-weighted global network. A policy network will be learned on top of the obtained global information, whose training objective is to capture the most task-relevant parts of the video directly in the 3D space formed by the frame height/width and the time. This is achieved by localizing a sequence of 3D cubes inside the original video. These smaller but informative video cubes will be progressively processed using a high-capacity but computationally more expensive local network for learning discriminative representations. This procedure is dramatically more efficient than processing the whole video due to the reduced size of the cubes. As a consequence, the computation is unevenly allocated across both spatial and temporal dimensions, resulting in considerable improvements in the overall efficiency with the maximally preserved accuracy.

It is noteworthy that the operation of selecting 3D cubes from the videos is not inherently differentiable. To enable the effective training of our AdaFocusV3 framework, we propose a gradient estimation algorithm for the policy network based on deep features. Our solution significantly outperforms the pixel-based counterpart proposed by AdaFocusV2 in our problem.

In addition, an intriguing property of AdaFocusV3 is that it naturally facilitates the adaptive inference. To be specific, since the informative video cubes are processed in a sequence, the inference procedure can be dynamically terminated once the model is able to produce sufficiently reliable predictions, such

that further redundant computation on relatively "easier" samples can be saved. This paradigm also allows AdaFocusV3 to adjust the computational cost online without additional training (by simply adapting the early-termination criterion). Such flexibility of our method enables it to make full use of the fluctuating computational resources or minimize the power consumption with varying performance targets. Both of them are the practical requirements of many real-world applications (*e.g.*, searching engines and mobile apps).

We validate the effectiveness of AdaFocusV3 on widely-used video recognition benchmarks. Empirical results show that AdaFocusV3 achieves a new state-of-the-art performance in terms of computational efficiency. For example, when obtaining the same accuracy, AdaFocusV3 has up to 2.6× less Multiply-Add operations compared to the recently proposed OCSampler [36] algorithm.

2 Related Works

Video recognition. Recent years have witnessed a significant improvement of the test accuracy on large-scale automatic video recognition benchmarks [3,29,30,21,40]. This remarkable progress is largely ascribed to the rapid development of video representation learning backbones [46,51,4,23,13,35,12,1]. The majority of these works focus on modeling the temporal relationships among different frames, which is a key challenge in video understanding. Towards this direction, a representative approach is leveraging the spatial-temporal information simultaneously by expanding the 2-D convolution to 3D space [46,4,23,12]. Another line of works propose to design specialized temporal-aware architectures on top of 2-D deep networks, such as adding recurrent networks [10,33,62], temporal feature averaging [51], and temporal channel shift [35,44,11,42]. Some other works model short-term and long-term temporal relationships separately using two-stream networks [15,14,13,20]. Besides, since processing videos with deep networks is computationally intensive, a variety of recent works have started to focus on developing efficient video recognition models [48,64,47,39,37,38].

Temporal dynamic networks. A straightforward approach for facilitating efficient video representation learning is to leverage the inherent temporal redundancy within videos. To be specific, not all video frames are equivalently relevant to a given task, such that ideally a model should dynamically identify less informative frames and allocate less computation to them correspondingly [22]. In the context of video recognition, a number of effective approaches have been proposed under this principle [61,58,18,32,57,41,19,31,45,36]. For example, OCSampler [36] proposes a one-step framework, learning to select task-relevant frames with reinforcement learning. VideoIQ [45] processes the frames using varying precision according to their importance. FrameExit [19] performs early-termination at test time after processing sufficiently informative frames. AdaFocusV3 is more general and flexible than these works since we simultaneously model both spatial and temporal redundancy.

Spatial dynamic networks. In addition to the temporal dimension, considerable spatial redundancy exists when processing video frames. As a matter

of fact, many works have revealed that deep networks can effectively extract representations from the image-based data via attending to a few task-relevant image regions [16,54,60,50]. Recently, the effectiveness of this paradigm has been validated in video recognition as well. As representative works, AdaFocus network V1&V2 [52,56] propose to infer the expensive high-capacity network only on a relatively small but informative patch of each frame, which is adaptively localized with a policy network, yielding a favorable accuracy-speed trade-off. An important advantage of the spatial-based approaches is that they are orthogonal to the temporal-adaptive methods. Nonetheless, in existing works, the spatial and temporal redundancy is usually modeled using independent algorithms with separate network architectures, and combined straightforwardly (*e.g.*, AdaFocusV1&V2 [52,56]). This paper assumes that such a naive implementation may lead to sub-optimal formulation, and proposes a unified AdaFocusV3 framework to consider spatial-temporal redundancy simultaneously, which achieves significantly higher computational efficiency.

3 Method

As aforementioned, existing efficient video recognition approaches usually consider the temporal or spatial redundancy of videos with another absent. We assume that such isolated modeling of the two types of redundancy may lead to sub-optimal formulation, and propose a unified AdaFocusV3 framework. AdaFocusV3 learns to perform spatial-temporal dynamic computation simultaneously, yielding significantly improved computational efficiency compared with the state-of-the-art baselines. In the following, we introduce the details of our method.

3.1 Overview

We first describe the inference and training pipelines of AdaFocusV3, laying the basis for the detailed introduction to the network architecture, the training algorithm and the implementation details.

Inference. We start by introducing the inference procedure of AdaFocusV3, which is shown in Figure 1. Given an input video $\mathbf{V} \in \mathbb{R}^{H \times W \times T}$ with T frames of $H \times W$ images (here we omit the RGB channels for simplicity), we first process it with a light-weighted global encoder f_{G}. The aim of this step is to obtain the coarse global information cheaply with low computational cost. Then the output features of f_{G} are fed into a policy network $\boldsymbol{\pi}$, which is trained to capture the most task-relevant parts of $\mathbf{V}$ for extracting finer representations. In specific, the outputs of $\boldsymbol{\pi}$ localize a sequence of 3D cubes $\{\tilde{\mathbf{V}}_1, \tilde{\mathbf{V}}_2, \dots\}$ with the size $H' \times W' \times T'$ $(H' < H, W' < W, T' < T)$, and these cubes will be processed by a local encoder f_{L}. Note that f_{L} is designed to be high-capacity, accurate and computationally more expensive. Considerable redundant computation can be saved by only activating f_{L} on top of the selected informative inputs with reduced size rather than the whole video.

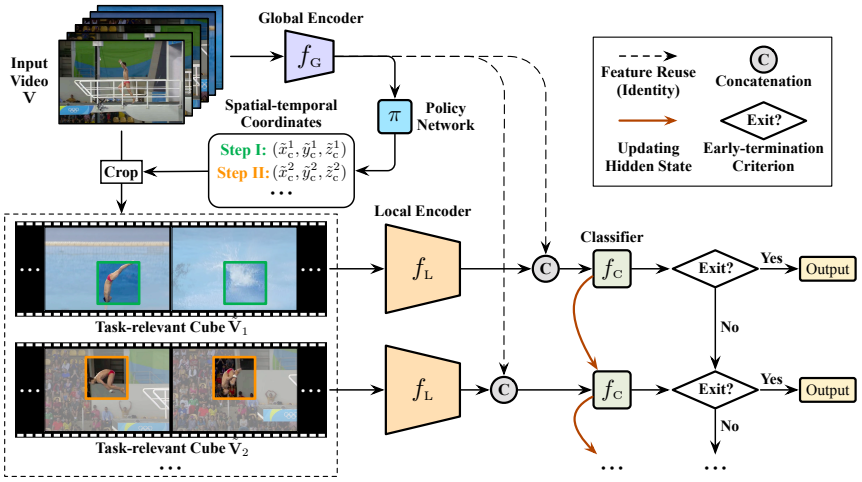

Fig. 1. Inference procedure of AdaFocusV3. The global encoder f_G first takes a quick glance at each input video in the holistic view. A policy network $\boldsymbol{\pi}$ is learned on top of f_G to capture the most task-relevant parts of the video with 3D spatial-temporal cubes. These cubes are processed by the high-capacity and accurate local encoder f_L for extracting discriminative deep features. A classifier $\boldsymbol{\pi}$ aggregates all the obtained information and dynamically outputs the prediction. Notably, the inference will be terminated once a convincing prediction (*e.g.*, with sufficiently low entropy or being adequately confident) has been produced.

At last, a classifier f_C aggregates the features of all the previous inputs to produce a prediction. Importantly, the contribution of $\{\tilde{\mathbf{V}}_1, \tilde{\mathbf{V}}_2, \dots\}$ to recognition are formulated to be descending in the sequence (via training, details described later). They are progressively processed by f_L and f_C, while a softmax prediction $\boldsymbol{p}_t$ will be retrieved from f_C after seeing every video cube $\tilde{\mathbf{V}}_t$. In other words, ideally, AdaFocusV3 always allocates the computation first to the most important video contents in terms of the task. The major insight behind this design is to facilitate the adaptive inference, *i.e.*, the inference can be terminated once $\boldsymbol{p}_t$ is sufficiently reliable and thus avoids further redundant computation. In the implementation, we adopt an entropy-based early-termination criterion (see Section 3.4 for details).

Training. AdaFocusV3 is trained by minimizing the sum of the loss corresponding to all the predictions from f_C, namely

$$\underset{f_G, f_L, f_C, \pi}{\text{minimize}} \quad \mathcal{L} = \mathop{\mathbb{E}}_{(\mathbf{V}, y) \in \mathcal{D}_{\text{train}}} \left[\sum\nolimits_t L_{\text{CE}}(\boldsymbol{p}_t, y) \right], \tag{1}$$

where y is the label of $\mathbf{V}$, $\mathcal{D}_{\text{train}}$ is the training set and $L_{\text{CE}}(\cdot)$ denotes the standard cross-entropy loss function. Intuitively, solving problem (1) will learn a $\boldsymbol{\pi}$ that enables the model to produce correct predictions with as fewer inputs as

possible. The selection policy is trained to usually find the most beneficial cubes for the current model at each step.

3.2 Network Architecture

Global encoder f_{G} and local encoder f_{L} are both deep networks (*e.g.*, ConvNets and vision Transformers) that map the input video frames to the deep feature space. However, f_{G} is exploited to take a quick glance at the videos, such that the policy network $\boldsymbol{\pi}$ can be activated to localize the informative video cubes. As a consequence, light-weighted architectures should be adopted. On the other hand, f_{L} is designed to extract accurate and discriminative representations from the selected important inputs. Therefore, it allows deploying computationally intensive and high-capacity models.

Policy network $\boldsymbol{\pi}$ receives the global feature maps produced by f_{G}, and determines the locations of the video cubes to attend to. Notably, here we assume all the frames of $\mathbf{V}$ are fed into f_{G} at the same time, since AdaFocusV3 seeks to reduce both the temporal and spatial redundancy simultaneously. In contrast, AdaFocusV2/V1 [52,56] by design processes the videos frame by frame. To this end, we do not follow their recurrent design of $\boldsymbol{\pi}$. On the contrary, we directly process the global features of the whole video, and obtain the centre coordinates of $\{\tilde{\mathbf{V}}_1, \tilde{\mathbf{V}}_2, \dots\}$ at one time.

Classifier f_{C} aggregates the information from $\{\tilde{\mathbf{V}}_1, \tilde{\mathbf{V}}_2, \dots\}$, and produces the final prediction correspondingly. The architecture of f_{C} may have various candidates, including recurrent networks [25,6,52], frame-wise prediction averaging [35,41,42], and accumulated feature pooling [19,56]. Besides, we adopt the efficient feature reuse mechanism proposed by [52,56], where the outputs of both f_{G} and f_{L} are leveraged by f_{C} (depicted using dashed lines in Figure 1).

3.3 Training Algorithm

Naive implementation. An obstacle towards solving problem (1) lies in that, cropping the cubes $\{\tilde{\mathbf{V}}_1, \tilde{\mathbf{V}}_2, \dots\}$ from the video $\mathbf{V}$ is not an inherently differentiable operation. Consequently, it is nontrivial to apply the standard back-propagation-based training algorithm. To address this issue, a straightforward approach is to leverage the interpolation-based cropping mechanism proposed by AdaFocusV2 [56]. Formally, given the centre coordinates $(\tilde{x}_{\mathrm{c}}^t, \tilde{y}_{\mathrm{c}}^t, \tilde{z}_{\mathrm{c}}^t)$ of $\tilde{\mathbf{V}}_t$, we can obtain $\tilde{\mathbf{V}}_t$ through trilinear interpolation. In this way, the value of any pixel $\tilde{v}_{i,j,k}^t$ in $\tilde{\mathbf{V}}_t$ is calculated as the weighted combination of its surrounded eight adjacent pixels in $\mathbf{V}$, *i.e.*,

$$\tilde{v}_{i,j,k}^t = \mathrm{Trilinear}\left(\mathbf{V}, (\tilde{x}_{i,j,k}^t, \tilde{y}_{i,j,k}^t, \tilde{z}_{i,j,k}^t)\right), \tag{2}$$

where $(\tilde{x}_{i,j,k}^t, \tilde{y}_{i,j,k}^t, \tilde{z}_{i,j,k}^t)$ corresponds to the horizontal, vertical and temporal coordinates of $\tilde{v}_{i,j,k}^t$ in $\mathbf{V}$.

In back-propagation, it is easy to obtain the gradients of the loss $\mathcal{L}$ with respect to $\tilde{v}^t_{i,j,k}$, *i.e.*, $\partial\mathcal{L}/\partial\tilde{v}^t_{i,j,k}$. With Eq. (2), we further have

$$\frac{\partial\mathcal{L}}{\partial\tilde{x}^t_{\mathrm{c}}}=\sum_{i,j,k}\frac{\partial\mathcal{L}}{\partial\tilde{v}^t_{i,j,k}}\frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{x}^t_{i,j,k}},\ \frac{\partial\mathcal{L}}{\partial\tilde{y}^t_{\mathrm{c}}}=\sum_{i,j,k}\frac{\partial\mathcal{L}}{\partial\tilde{v}^t_{i,j,k}}\frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{y}^t_{i,j,k}},\ \frac{\partial\mathcal{L}}{\partial\tilde{z}^t_{\mathrm{c}}}=\sum_{i,j,k}\frac{\partial\mathcal{L}}{\partial\tilde{v}^t_{i,j,k}}\frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{z}^t_{i,j,k}}. \quad (3)$$

Note that Eq. (3) leverages the fact that the geometric relationship between $(\tilde{x}^t_{i,j,k},\tilde{y}^t_{i,j,k},\tilde{z}^t_{i,j,k})$ and $(\tilde{x}^t_{\mathrm{c}},\tilde{y}^t_{\mathrm{c}},\tilde{z}^t_{\mathrm{c}})$ is always fixed once the size of $\tilde{\mathbf{V}}_t$ is fixed, namely we have

$$\frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{x}^t_{\mathrm{c}}}=\frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{x}^t_{i,j,k}},\ \frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{y}^t_{\mathrm{c}}}=\frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{y}^t_{i,j,k}},\ \frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{z}^t_{\mathrm{c}}}=\frac{\partial\tilde{v}^t_{i,j,k}}{\partial\tilde{z}^t_{i,j,k}}. \quad (4)$$

Given that $\tilde{x}^t_{\mathrm{c}}$, $\tilde{y}^t_{\mathrm{c}}$ and $\tilde{z}^t_{\mathrm{c}}$ are the outputs of $\boldsymbol{\pi}$, the regular back-propagation are capable of proceeding through Eq. (3) to compute the gradients of all the parameters and update the model.

In essence, calculating the gradients with Eqs. (2-4) follows the following logic. We change the location of $\tilde{\mathbf{V}}_t$ in an infinitesimal quantity, and measure the consequent changes on the value of each pixel in $\tilde{\mathbf{V}}_t$ via Eq. (2). These changes affect the inputs of f_{L}, and accordingly, have an impact on the final optimization objective.

Limitations of pixel-based gradient estimation. Although the aforementioned procedure enables the back-propagation of gradients, it suffers from some major issues. First, the change of the contents of $\tilde{\mathbf{V}}_t$ is achieved by varying the pixel values, each of which is determined only by few adjacent pixels in $\mathbf{V}$. Such a limited source of information may be too local to reflect the semantic-level changes when the location of $\tilde{\mathbf{V}}_t$ varies. However, during training, the model should ideally be informed with how the semantical contents of $\tilde{\mathbf{V}}_t$ will change with the different outputs of $\boldsymbol{\pi}$, as we hope to localize the most informative parts of the original videos. Second, performing interpolation along the temporal dimension amounts to mixing the adjacent frames in the pixel space, which is empirically observed to degrade the generalization performance of the model. We tentatively attribute this to the violation of the i.i.d. assumption, since the frames are not mixed at test time.

Estimating gradients with deep features. To alleviate the problems caused by the pixel-based gradient estimation, we propose a novel deep feature based approach to guide the training of the policy network $\boldsymbol{\pi}$. Our main motivation here is to enable the outputs of $\boldsymbol{\pi}$ to have a direct influence on the feature maps that are extracted by f_{L}. These deep representations are known to excel at capturing the task-relevant semantics of the inputs [2,49,55]. Once the gradients can explicitly tell $\boldsymbol{\pi}$ how the deep features corresponding to $\tilde{\mathbf{V}}_t$ will be affected by its outputs, $\boldsymbol{\pi}$ will receive direct supervision signals on whether $\tilde{\mathbf{V}}_t$ contains valuable contents in terms of the recognition task.

To implement this idea, assume that $\boldsymbol{e}_t \in \mathbb{R}^{H^{\mathrm{f}}\times W^{\mathrm{f}}\times T^{\mathrm{f}}}$ denotes the feature maps of $\tilde{\mathbf{V}}_t$ produced by f_{L}. During training, we crop a slightly larger cube $\tilde{\mathbf{V}}'_t$ at the location of $\tilde{\mathbf{V}}_t$ from the original video $\mathbf{V}$, such that the size of the feature

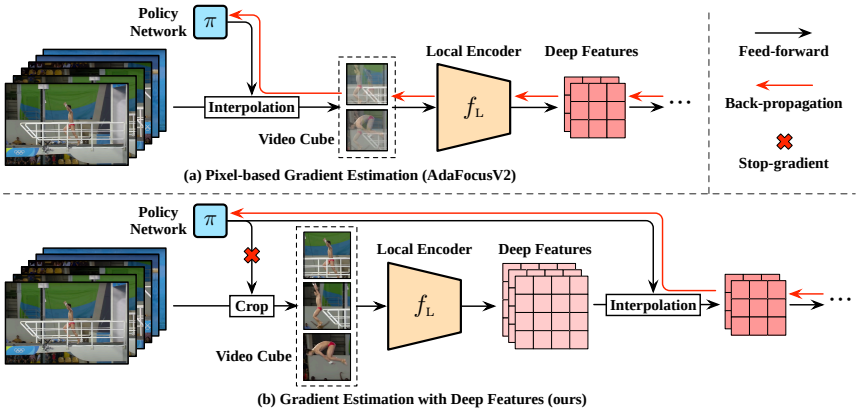

Fig. 2. Comparisons of the gradient estimation with image pixels and deep features. The back-propagation process of the policy network $\boldsymbol{\pi}$ is illustrated with **red arrows**. AdaFocusV3 proposes to guide the learning of $\boldsymbol{\pi}$ with deep features. In comparison with the pixel-based approach proposed in AdaFocusV2, our method not only provides more direct semantic-level supervision signals for $\boldsymbol{\pi}$, but also avoids the issue that the adjacent frames are mixed in the pixel space during training.

maps $\boldsymbol{e}'_t$ corresponding to $\tilde{\mathbf{V}}'_t$ will exactly be $(H^{\mathrm{f}}{+}1){\times}(W^{\mathrm{f}}{+}1){\times}(T^{\mathrm{f}}{+}1)$. Note that here we deactivate the gradient computation and hence the back-propagation through Eqs. (2-4) will not happen. Then we obtain $\boldsymbol{e}_t$ by performing trilinear interpolation at the centre of $\boldsymbol{e}'_t$. Formally, the value of any element $e^t_{i,j,k}$ in $\boldsymbol{e}_t$ can be solved by

$$e^t_{i,j,k} = \text{Trilinear}\left(\boldsymbol{e}'_t, (x(e^t_{i,j,k}), y(e^t_{i,j,k}), z(e^t_{i,j,k}))\right). \tag{5}$$

Herein, $x(e^t_{i,j,k})$, $y(e^t_{i,j,k})$ and $z(e^t_{i,j,k})$ are the coordinates in $\boldsymbol{e}'_t$ corresponding to $e^t_{i,j,k}$. They are given by:

$$(x(e^t_{i,j,k}), y(e^t_{i,j,k}), z(e^t_{i,j,k})) = (\tilde{x}^t_{\mathrm{c}}, \tilde{y}^t_{\mathrm{c}}, \tilde{z}^t_{\mathrm{c}}) - \text{StopGradient}(\tilde{x}^t_{\mathrm{c}}, \tilde{y}^t_{\mathrm{c}}, \tilde{z}^t_{\mathrm{c}}) + \boldsymbol{o}_{i,j,k}, \tag{6}$$

where $\boldsymbol{o}_{i,j,k}$ is a vector from $(0,0,0)$ to the location of $e^t_{i,j,k}$, and it is only conditioned on i,j,k and the shape of $e^t_{i,j,k}$. When calculating the gradients with Eq. (6), the infinitesimal change on the outputs of $\boldsymbol{\pi}$ (*i.e.*, $\tilde{x}^t_{\mathrm{c}}, \tilde{y}^t_{\mathrm{c}}, \tilde{z}^t_{\mathrm{c}}$) will directly act on the feature maps $\boldsymbol{e}_t$. To sum up, in the feed-forward process, we obtain the enlarged cube $\tilde{\mathbf{V}}'_t$ in the pixel space based on outputs of $\boldsymbol{\pi}$, while for back-propagation, the supervision signals for $\boldsymbol{\pi}$ come from the semantic-level deep feature space. In addition, it is noteworthy that only the training process is modified. At test time, we crop $\tilde{\mathbf{V}}_t$ and activate f_{L} as stated in Section 3.1.

3.4 Implementation Details

Adaptive inference. As mentioned in Section 3.1, the inference procedure can be terminated once the model is already able to produce a sufficiently reliable prediction. Consequently, the computation is unevenly allocated across "easy" and "hard" videos, improving the overall efficiency. In implementation, after processing the inputs $\{\tilde{\mathbf{V}}_1, \ldots, \tilde{\mathbf{V}}_2\}$, the classifier f_{C} integrates all the acquired information and outputs a softmax prediction $\boldsymbol{p}_t$. We compare the entropy of $\boldsymbol{p}_t$ (*i.e.*, $-\sum_j p_{ij} \log p_{ij}$) with a threshold η_t. The inference will be terminated with $\boldsymbol{p}_t$ if we have $-\sum_j p_{ij} \log p_{ij} \leq \eta_t$. The thresholds $\{\eta_1, \eta_2, \ldots\}$ can be solved on the validation set $\mathcal{D}_{\mathrm{val}}$, under the principle of maximizing the accuracy with a limited computational budget $B > 0$:

$$\underset{\eta_1, \eta_2, \ldots}{\text{maximize}} \ \text{Accuracy}(\eta_1, \eta_2, \ldots | \mathcal{D}_{\mathrm{val}}) \quad \text{s.t. FLOPs}(\eta_1, \eta_2, \ldots | \mathcal{D}_{\mathrm{val}}) \leq B. \tag{7}$$

As a matter of fact, by varying the value of B, a group of $\{\eta_1, \eta_2, \ldots\}$ can be obtained, corresponding to a variety of computational constraints. The computational cost of our proposed AdaFocusV3 framework can be online adjusted without additional training by simply adapting these termination criterion. In this paper, we solve problem (7) following the methodology proposed in [26,54,53].

Glance step. As aforementioned, AdaFocusV3 first takes a glance at the input video with the light-weighted global encoder f_{G}. Although being cheap and coarse, the global representations obtained from f_{G} may have learned certain discriminative features of the inputs. To this end, in addition to feeding them into the policy network $\boldsymbol{\pi}$, we also activate classifier f_{C} to produce a quick prediction $\boldsymbol{p}_0$ only using this global information, aiming to facilitate efficient feature reuse.

4 Experiment

In this section, we present a comprehensive experimental comparison between our proposed AdaFocusV3 and state-of-the-art efficient video recognition frameworks. AdaFocusV3 significantly outperforms these competitive baselines in terms of computational efficiency. We also demonstrate that AdaFocusV3 can be deployed on the basis of recently processed light-weighted deep networks, and further improve their efficiency. The analytical results including the ablation study and visualization are provided to give additional insights into our method.

Datasets. Our experiments are based on six large-scale video recognition benchmark datasets, *i.e.*, ActivityNet [3], FCVID [29], Mini-Kinetics [30,59], Something-Something (Sth-Sth) V1&V2 [21] and Diving48 [34]. The official training-validation split is adopted for all of them. Note that these datasets are widely used in the experiments of a considerable number of recently proposed baselines. We select them for a reasonable comparison with current state-of-the-art results. Due to the spatial limitations, the detailed introductions on datasets and data pre-processing are deferred to Appendix A.

Metrics. The offline video recognition is considered, where a single prediction is required for each video. Following the common practice on the aforementioned datasets, we evaluate the performance of different methods on ActivityNet and FCVID via mean average precision (mAP), while the top-1 accuracy is adopted on Mini-Kinetics, Sth-Sth V1&V2 and Diving48. In addition, we measure the theoretical computational cost of the models using the average number of multiply-add operations required for processing a video (*i.e.*, GFLOPs/Video). To compare the practical inference speed of different methods, we report the maximal throughput (Videos/s) when a given hardware (*e.g.*, GPU) is saturated.

4.1 Comparisons with State-of-the-art Baselines

Baselines. We first compare AdaFocusV3 with a variety of recently proposed approaches that focus on improving the efficiency of video recognition. The results on ActivityNet, FCVID and Mini-Kinetics are provided. In addition to the previous versions of AdaFocus, the following baselines are included, *i.e.*, LiteEval [59], SCSampler [32], ListenToLook [18], AR-Net [41], AdaFrame [58], VideoIQ [45], and OCSampler [36]. An introduction of them is presented in Appendix B.

Implementation details. For a fair comparison, the design of backbone networks follows the common setups of the baselines. A MobileNet-V2 [43] and a ResNet-50 [24] are used as the global encoder f_{G} and local encoder f_{L} in AdaFocusV3. The architecture of the classifier f_{C} follows from [19,56]. In addition, given that the videos in ActivityNet, FCVID and Mini-Kinetics are relatively long, previous works have revealed that a single uniformly sampled frame is sufficient to represent the information within the adjacent video clip [58,19,36]. Hence, we set the cube size of AdaFocusV3 to be $96 \times 96 \times 1$ and $128 \times 128 \times 1$, which enables our method to flexibly perform spatial-temporal dynamic computation in a frame-wise fashion. The maximum number of video cubes is set to 18. During inference, we remove the frames which have been included by previous cubes. The training configurations can be found in Appendix C.

Main results. We report the performance of AdaFocusV3 and the baselines in Table 1. The results of our method are presented when its accuracy or computational cost reaches the best records of all the baselines. It is clear that AdaFocusV3 outperforms other efficient video recognition frameworks dramatically. For example, it improves the mAP by $\sim 2.6-3.2\%$ compared to the competitive OCSampler [36] algorithm on ActivityNet and FCVID with the same GFLOPs. A more comprehensive comparison is depicted in Figure 3, where we vary the computational budget and plot the ActivityNet mAP v.s. GFLOPs relationships of both AdaFocusV3 and the variants of baselines. The two black curves correspond to adopting the two sizes of video cubes. One can observed that AdaFocusV3 yields a considerably better accuracy-efficiency trade-off consistently. It reduces the demands of computation by $\sim 1.6-2.6\times$ on top of the strongest baselines without sacrificing mAP.

Comparisons of the unified and separate modeling of spatial and temporal redundancy. We further empirically validate whether a unified frame-

Table 1. Performance of AdaFocusV3 and the baselines on three benchmark datasets. The comparisons are based on the same backbones, where MN2/RN denotes MobileNet-V2/ResNet and † represents adding TSM [35]. GFLOPs refer to the average computational cost for processing each video. The best two results are **bold-faced** and underlined, respectively. Blue numbers are obtained on top of the strongest baselines.

Methods	Published on	Backbones	ActivityNet		FCVID		Mini-Kinetics	
			mAP	GFLOPs	mAP	GFLOPs	Top-1 Acc.	GFLOPs
LiteEval [59]	NeurIPS'19	MN2+RN	72.7%	95.1	80.0%	94.3	61.0%	99.0
SCSampler [32]	ICCV'19	MN2+RN	72.9%	42.0	81.0%	42.0	70.8%	42.0
ListenToLook [18]	CVPR'20	MN2+RN	72.3%	81.4	–	–	–	–
AR-Net [41]	ECCV'20	MN2+RN	73.8%	33.5	81.3%	35.1	71.7%	32.0
AdaFrame [58]	T-PAMI'21	MN2+RN	71.5%	79.0	80.2%	75.1	–	–
VideoIQ [45]	ICCV'21	MN2+RN	74.8%	28.1	82.7%	27.0	72.3%	20.4
OCSampler [36]	CVPR'22	MN2†+RN	76.9%	21.7	82.7%	26.8	72.9%	17.5
AdaFocusV3	–	MN2+RN	76.9%	**10.9**$_{\downarrow 2.0\times}$	82.7%	**7.8**$_{\downarrow 3.4\times}$	72.9%	**8.6**$_{\downarrow 2.0\times}$
	–	MN2+RN	**79.5%**$_{\uparrow 2.6\%}$	21.7	**85.9%**$_{\uparrow 3.2\%}$	26.8	**75.0%**$_{\uparrow 2.1\%}$	17.5

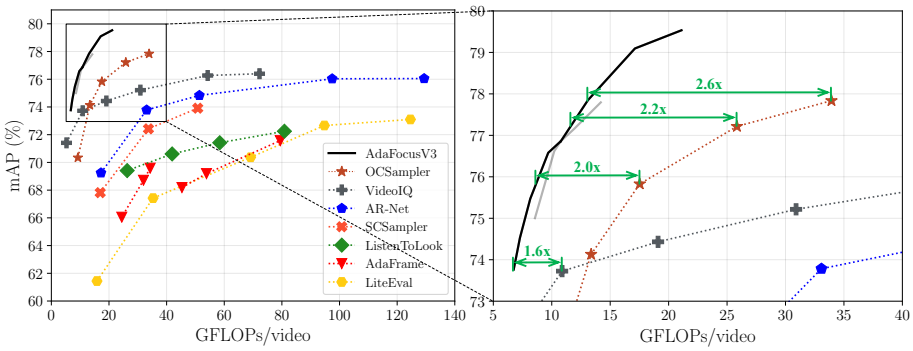

Fig. 3. AdaFocusV3 v.s. state-of-the-art baselines on ActivityNet in terms of computational efficiency. Note that the computational cost of AdaFocusV3 can be adjusted online (*i.e.*, switching within each black curve without additional training).

work for spatial-temporal dynamic computation indeed outperforms the isolatedly-modeling counterpart. This is achieved by comparing AdaFocusV3 with AdaFocusV2+ [56]. The latter has the same network architecture and training procedure as AdaFocusV3, while the only difference lies in that AdaFocusV2+ models spatial and temporal redundancy independently in isolated modules. The results in Table 2 illustrate that our proposed unified framework efficiently reduce the computational cost (by $\sim 1.3\times$) with preserved accuracy.

4.2 Deploying on Top of Light-weighted Models

Setups. In this subsection, we implement our framework on top of a representative efficient video analysis network, ConvNets with temporal shift module

Table 2. Comparisons of AdaFocusV3 and AdaFocusV2+. The latter differs from AdaFocusV3 only in that it models the spatial and temporal redundancy independently in isolated modules. We report the computational cost of the two methods when they reach the same accuracy.

Dataset	mAP	Computational Cost (GFLOPs/video)	
		AdaFocusV2+	AdaFocusV3 (ours)
ActivityNet	77.0%	14.0	**11.1** (↓1.3×)
	78.0%	17.6	**13.6** (↓1.3×)
	79.0%	24.4	**16.8** (↓1.5×)
FCVID	83.0%	10.8	**8.3** (↓1.3×)
	84.0%	14.2	**10.5** (↓1.4×)
	85.0%	24.4	**14.4** (↓1.7×)

Table 3. Performance of AdaFocusV3 on top of TSM. MN2, R18/R34/R50 and BN-Inc. refer to MobileNet-V2, ResNet-18/34/50 and BN-Inception, respectively. TSM+ denotes the augmented TSM baseline with the same backbone network architecture as our method. The best results are **bold-faced**.

Method	Backbones	Sth-Sth V1		Sth-Sth V2		Throughput (Sth-Sth V1)
		Top-1 Acc.	GFLOPs	Top-1 Acc.	GFLOPs	(NVIDIA 3090 GPU, bs=128)
TSN [51]	R50	19.7%	33.2	27.8%	33.2	-
AR-Net [41]	MN2+R18/34/50	18.9%	41.4	-	-	-
$\text{TRN}_{\text{RGB/Flow}}$ [63]	BN-Inc.	42.0%	32.0	55.5%	32.0	-
ECO [64]	BN-Inc.+3DR18	39.6%	32.0	-	-	-
STM [28]	R50	47.5%	33.3	-	-	-
TSM [35]	R50	46.1%	32.7	59.1%	32.7	162.7 Videos/s
AdaFuse-TSM [42]	R50	46.8%	31.5	59.8%	31.3	-
TSM+ [35]	MN2+R50	47.0%	35.1	59.6%	35.1	123.0 Videos/s
AdaFocusV2	MN2+R50	47.0%	18.5 (↓1.9×)	59.6%	18.5 (↓1.9×)	197.0 Videos/s (↑1.6×)
AdaFocusV3	MN2+R50	47.0%	**14.0** (↓**2.5**×)	59.6%	**15.4** (↓**2.3**×)	**234.0 Videos/s** (↑**1.9**×)

Table 4. AdaFocusV3 on top of TSM with varying sizes of video cubes. MN2/R50 denotes MobileNetV2/ResNet50.

Method	Backbones	Size of 3D Video Cubes	Sth-Sth V1		Sth-Sth V2		Diving48	
			Acc.	GFLOPs	Acc.	GFLOPs	Acc.	GFLOPs
TSM [35]	R50	–	46.1%	32.7	59.1%	32.7	77.4%	32.7
TSM+ [35]	MN2+R50	$224^2\times16$	47.0%	35.1	59.6%	35.1	80.4%	35.1
AdaFocusV3	MN2+R50	$128^2\times8$	46.1%	14.0	58.5%	15.4	79.3%	6.2
AdaFocusV3	MN2+R50	$128^2\times12$	**47.0%**	**14.0**	**59.6%**	**15.4**	**80.4%**	**6.2**
AdaFocusV3	MN2+R50	$128^2\times16$	47.0%	15.9	59.6%	17.2	80.4%	7.7

(TSM) [35], to demonstrate its effectiveness on recently proposed light-weighted models. The network architecture is not changed except for adding TSM to f_{G} and f_{L}. We uniformly sample 8 video frames for f_{G}. Unless otherwise specified, the size of the video cubes for f_{L} is set to be $128\times128\times12$, while the maximum number of cubes is 2. The training configurations can be found in Appendix C.

Results on Sth-Sth V1&V2 are presented in Table 3. For fair comparisons, following [56], we augment the vanilla TSM by adopting the same two backbone networks as ours, which is named as TSM+. It can be observed that AdaFocusV3 saves the computational cost by up to 2.5× compared with TSM+ when reaching the same Top-1 Acc. We also test the practical efficiency of our method on GPU

Table 5. Ablation study on the learned policy and training algorithm.

Gradient Estimation with Deep Features	Learned Temporal-dynamic Policy	Learned Spatial-dynamic Policy	ActivityNet mAP after processing t video cubes (*i.e.*, with $\boldsymbol{p}_t$) t=1	t=2	t=4	t=8
–	✗ (random)	✗ (random)	44.6%	57.6%	67.4%	73.7%
✓	✓	✗ (random)	50.1%	61.3%	69.4%	74.6%
✓	✗ (random)	✓	45.5%	58.2%	68.5%	74.6%
✗ (pixel-based)	✓	✓	50.6%	61.0%	69.5%	75.0%
✓	✓	✓	**51.9%**	**62.3%**	**70.6%**	**75.4%**

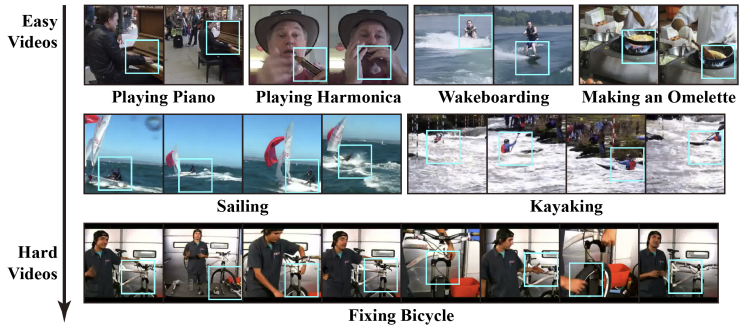

Fig. 4. Visualization results on ActivityNet. Easy and hard videos refer to the samples that need a small or large number of video cubes for being correctly recognized.

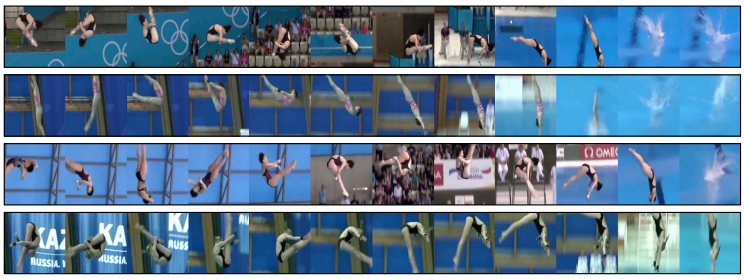

Fig. 5. Visualization results on Diving48. We present the examples of the task-relevant 3D video cubes localized by AdaFocusV3.

devices, where the videos are fed into the model in batches, and the samples that meet the early-termination criterion will be output at each exit. The results indicate that the real speedup of AdaFocusV3 is significant as well.

Results with varying cube sizes are summarized in Table 4. We find that reducing the cube size in the temporal dimension ($128^2 \times 12 \rightarrow 128^2 \times 8$) significantly degrades the accuracy on Sth-Sth/Diving48. Also interestingly, larger

cubes like $128^2 \times 16$ may weaken the temporal adaptiveness, yielding inferior computational efficiency. Another interesting observation is that on Diving48, AdaFocusV3 dramatically reduces the computational cost (by 5.7×) on top of the baseline (TSM+) with the same accuracy. This phenomenon may be explained by that AdaFocusV3 will be more effective in less biased scenarios, where the background is less informative, and thus it is more sensible to attend to the task-relevant foreground parts of videos.

4.3 Analytical Results

Ablation study. In Table 5, we test ablating the cube selection policy and the gradient estimation technique in AdaFocusV3. The results on ActivityNet with the cube size of $128^2 \times 1$ are presented. For a clean comparison, here we deactivate the glance step as well as the early-termination algorithm, and report the mAP corresponding to processing a fixed number of video cubes. One can observe that our learned policy considerably outperforms the random baselines in terms of either spatial or temporal dimensions. In addition, our gradient estimation technique based on deep features significantly outperforms the pixel-based counterpart proposed by AdaFocusV2 [56].

Visualization. In Figure 4, the visualization on ActivityNet with the cube size of $96^2 \times 1$ is presented. The blue boxes indicate the video cubes localized by AdaFocusV3. It is shown that our method adaptively attends to the task-relevant parts of the video, such as the person, the piano and the bicycle. Figure 5 shows the examples of video cubes on Diving48. AdaFocusV3 can adaptively capture the actions of the diving athletes – the crucial contents for recognition.

5 Conclusion

In this paper, we proposed an AdaFocusV3 framework which enables the unified formulation of the efficient spatial-temporal dynamic computation in video recognition. AdaFocusV3 is trained to adaptively identify and attend to the most task-relevant video cubes in the 3D space formed by frame height/width and the time. The number of these cubes are determined conditioned on each sample under the dynamic early-termination paradigm. Extensive empirical results on six benchmark datasets validate the state-of-the-art performance of our model in terms of computational efficiency. Our work may open new avenues for the simultaneous modeling of spatial and temporal redundancy in video recognition.

Acknowledgements

This work is supported in part by National Key R&D Program of China (2020AAA0105200), the National Natural Science Foundation of China under Grants 62022048, Guoqiang Institute of Tsinghua University and Beijing Academy of Artificial Intelligence. We also appreciate the generous donation of computing resources by High-Flyer AI.

References

1. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. arXiv preprint arXiv:2103.15691 (2021) 1, 3
2. Bengio, Y., Mesnil, G., Dauphin, Y., Rifai, S.: Better mixing via deep representations. In: ICML. pp. 552–560. PMLR (2013) 7
3. Caba Heilbron, F., Escorcia, V., Ghanem, B., Carlos Niebles, J.: Activitynet: A large-scale video benchmark for human activity understanding. In: CVPR. pp. 961–970 (2015) 3, 9
4. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: CVPR. pp. 6299–6308 (2017) 1, 3
5. Chen, J., Li, K., Deng, Q., Li, K., Philip, S.Y.: Distributed deep learning model for intelligent video surveillance systems with edge computing. IEEE Transactions on Industrial Informatics (2019) 2
6. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: EMNLP. pp. 1724–1734. Association for Computational Linguistics, Doha, Qatar (Oct 2014). https://doi.org/10.3115/v1/D14-1179, https://www.aclweb.org/anthology/D14-1179 6
7. Collins, R.T., Lipton, A.J., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O., Burt, P., et al.: A system for video surveillance and monitoring. VSAM final report **2000**(1-68), 1 (2000) 2
8. Davidson, J., Liebald, B., Liu, J., Nandy, P., Van Vleet, T., Gargi, U., Gupta, S., He, Y., Lambert, M., Livingston, B., et al.: The youtube video recommendation system. In: Proceedings of the fourth ACM conference on Recommender systems. pp. 293–296 (2010) 2
9. Deldjoo, Y., Elahi, M., Cremonesi, P., Garzotto, F., Piazzolla, P., Quadrana, M.: Content-based video recommendation system based on stylistic visual features. Journal on Data Semantics **5**(2), 99–113 (2016) 2
10. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: CVPR. pp. 2625–2634 (2015) 3
11. Fan, L., Buch, S., Wang, G., Cao, R., Zhu, Y., Niebles, J.C., Fei-Fei, L.: Rubiksnet: Learnable 3d-shift for efficient video action recognition. In: ECCV. pp. 505–521. Springer (2020) 3
12. Feichtenhofer, C.: X3d: Expanding architectures for efficient video recognition. In: CVPR. pp. 203–213 (2020) 1, 3
13. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: ICCV. pp. 6202–6211 (2019) 1, 3
14. Feichtenhofer, C., Pinz, A., Wildes, R.P.: Spatiotemporal multiplier networks for video action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4768–4777 (2017) 3
15. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: CVPR. pp. 1933–1941 (2016) 1, 3
16. Figurnov, M., Collins, M.D., Zhu, Y., Zhang, L., Huang, J., Vetrov, D., Salakhutdinov, R.: Spatially adaptive computation time for residual networks. In: CVPR. pp. 1039–1048 (2017) 4
17. Gao, J., Zhang, T., Xu, C.: A unified personalized video recommendation via dynamic recurrent neural networks. In: ACM MM. pp. 127–135 (2017) 2

18. Gao, R., Oh, T.H., Grauman, K., Torresani, L.: Listen to look: Action recognition by previewing audio. In: CVPR. pp. 10457–10467 (2020) 2, 3, 10, 11
19. Ghodrati, A., Bejnordi, B.E., Habibian, A.: Frameexit: Conditional early exiting for efficient video recognition. In: CVPR. pp. 15608–15618 (2021) 3, 6, 10
20. Gong, X., Wang, H., Shou, M.Z., Feiszli, M., Wang, Z., Yan, Z.: Searching for two-stream models in multivariate space for video recognition. In: ICCV. pp. 8033–8042 (2021) 3
21. Goyal, R., Ebrahimi Kahou, S., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., et al.: The "something something" video database for learning and evaluating visual common sense. In: ICCV. pp. 5842–5850 (2017) 3, 9
22. Han, Y., Huang, G., Song, S., Yang, L., Wang, H., Wang, Y.: Dynamic neural networks: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2021) 3
23. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In: CVPR. pp. 6546–6555 (2018) 1, 3
24. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016) 10
25. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997) 6
26. Huang, G., Chen, D., Li, T., Wu, F., van der Maaten, L., Weinberger, K.Q.: Multi-scale dense networks for resource efficient image classification. In: ICLR (2018) 9
27. Ikizler, N., Forsyth, D.: Searching video for complex activities with finite state models. In: CVPR. pp. 1–8. IEEE (2007) 2
28. Jiang, B., Wang, M., Gan, W., Wu, W., Yan, J.: Stm: Spatiotemporal and motion encoding for action recognition. In: ICCV. pp. 2000–2009 (2019) 12
29. Jiang, Y.G., Wu, Z., Wang, J., Xue, X., Chang, S.F.: Exploiting feature and class relationships in video categorization with regularized deep neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**(2), 352–364 (2018) 3, 9
30. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017) 3, 9
31. Kim, H., Jain, M., Lee, J.T., Yun, S., Porikli, F.: Efficient action recognition via dynamic knowledge propagation. In: ICCV. pp. 13719–13728 (2021) 3
32. Korbar, B., Tran, D., Torresani, L.: Scsampler: Sampling salient clips from video for efficient action recognition. In: ICCV. pp. 6232–6242 (2019) 2, 3, 10, 11
33. Li, D., Qiu, Z., Dai, Q., Yao, T., Mei, T.: Recurrent tubelet proposal and recognition networks for action detection. In: ECCV. pp. 303–318 (2018) 3
34. Li, Y., Li, Y., Vasconcelos, N.: Resound: Towards action recognition without representation bias. In: ECCV. pp. 513–528 (2018) 9
35. Lin, J., Gan, C., Han, S.: Tsm: Temporal shift module for efficient video understanding. In: ICCV. pp. 7083–7093 (2019) 3, 6, 11, 12
36. Lin, J., Duan, H., Chen, K., Lin, D., Wang, L.: Ocsampler: Compressing videos to one clip with single-step sampling. In: CVPR (2022) 2, 3, 10, 11
37. Liu, Z., Luo, D., Wang, Y., Wang, L., Tai, Y., Wang, C., Li, J., Huang, F., Lu, T.: Teinet: Towards an efficient architecture for video recognition. In: AAAI. pp. 11669–11676 (2020) 3
38. Liu, Z., Wang, L., Wu, W., Qian, C., Lu, T.: Tam: Temporal adaptive module for video recognition. In: ICCV. pp. 13708–13718 (2021) 3

39. Luo, C., Yuille, A.L.: Grouped spatial-temporal aggregation for efficient action recognition. In: ICCV. pp. 5512–5521 (2019) 3
40. Materzynska, J., Berger, G., Bax, I., Memisevic, R.: The jester dataset: A large-scale video dataset of human gestures. In: ICCVW (2019) 3
41. Meng, Y., Lin, C.C., Panda, R., Sattigeri, P., Karlinsky, L., Oliva, A., Saenko, K., Feris, R.: Ar-net: Adaptive frame resolution for efficient action recognition. In: ECCV. pp. 86–104. Springer (2020) 2, 3, 6, 10, 11, 12
42. Meng, Y., Panda, R., Lin, C.C., Sattigeri, P., Karlinsky, L., Saenko, K., Oliva, A., Feris, R.: Adafuse: Adaptive temporal fusion network for efficient action recognition. In: ICLR (2021) 3, 6, 12
43. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: CVPR. pp. 4510–4520 (2018) 10
44. Sudhakaran, S., Escalera, S., Lanz, O.: Gate-shift networks for video action recognition. In: CVPR. pp. 1102–1111 (2020) 3
45. Sun, X., Panda, R., Chen, C.F.R., Oliva, A., Feris, R., Saenko, K.: Dynamic network quantization for efficient video inference. In: ICCV. pp. 7375–7385 (2021) 2, 3, 10, 11
46. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV. pp. 4489–4497 (2015) 1, 3
47. Tran, D., Wang, H., Torresani, L., Feiszli, M.: Video classification with channel-separated convolutional networks. In: ICCV. pp. 5552–5561 (2019) 3
48. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: CVPR. pp. 6450–6459 (2018) 3
49. Upchurch, P., Gardner, J., Pleiss, G., Pless, R., Snavely, N., Bala, K., Weinberger, K.: Deep feature interpolation for image content changes. In: CVPR. pp. 7064–7073 (2017) 7
50. Verelst, T., Tuytelaars, T.: Dynamic convolutions: Exploiting spatial sparsity for faster inference. In: CVPR. pp. 2320–2329 (2020) 4
51. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: ECCV. pp. 20–36. Springer (2016) 3, 12
52. Wang, Y., Chen, Z., Jiang, H., Song, S., Han, Y., Huang, G.: Adaptive focus for efficient video recognition. In: ICCV (October 2021) 2, 4, 6
53. Wang, Y., Huang, R., Song, S., Huang, Z., Huang, G.: Not all images are worth 16x16 words: Dynamic transformers for efficient image recognition. In: NeurIPS (2021) 9
54. Wang, Y., Lv, K., Huang, R., Song, S., Yang, L., Huang, G.: Glance and focus: a dynamic approach to reducing spatial redundancy in image classification. In: NeurIPS (2020) 4, 9
55. Wang, Y., Pan, X., Song, S., Zhang, H., Huang, G., Wu, C.: Implicit semantic data augmentation for deep networks. NeurIPS **32** (2019) 7
56. Wang, Y., Yue, Y., Lin, Y., Jiang, H., Lai, Z., Kulikov, V., Orlov, N., Shi, H., Huang, G.: Adafocus v2: End-to-end training of spatial dynamic networks for video recognition. In: CVPR (2022) 2, 4, 6, 10, 11, 13, 14
57. Wu, W., He, D., Tan, X., Chen, S., Wen, S.: Multi-agent reinforcement learning based frame sampling for effective untrimmed video recognition. In: ICCV. pp. 6222–6231 (2019a) 3

58. Wu, Z., Li, H., Xiong, C., Jiang, Y.G., Davis, L.S.: A dynamic frame selection framework for fast video recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020b) 2, 3, 10, 11
59. Wu, Z., Xiong, C., Jiang, Y.G., Davis, L.S.: Liteeval: A coarse-to-fine framework for resource efficient video recognition. In: NeurIPS (2019b) 9, 10, 11
60. Xie, Z., Zhang, Z., Zhu, X., Huang, G., Lin, S.: Spatially adaptive inference with stochastic feature sampling and interpolation. In: ECCV. pp. 531–548. Springer (2020) 4
61. Yeung, S., Russakovsky, O., Mori, G., Fei-Fei, L.: End-to-end learning of action detection from frame glimpses in videos. In: CVPR. pp. 2678–2687 (2016) 2, 3
62. Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: Deep networks for video classification. In: CVPR. pp. 4694–4702 (2015) 3
63. Zhou, B., Andonian, A., Oliva, A., Torralba, A.: Temporal relational reasoning in videos. In: ECCV. pp. 803–818 (2018) 12
64. Zolfaghari, M., Singh, K., Brox, T.: Eco: Efficient convolutional network for online video understanding. In: Proceedings of the European conference on computer vision (ECCV). pp. 695–712 (2018) 3, 12